NORTHERN ARIZONA UNIVERSITY College of Engineering, Forestry, and Natural Sciences

Group:	James Beasley, Charles Beck, Charles Duso, Alexander Grzesiak, Erik Strauss
Project Title:	Boston University - Microfluid Experimentation Data Generator
Deliverable:	D.5 Internal Design
Course:	CS386 – Spring 2017
Instructor:	Professor Gerosa
Github:	https://github.com/TheAwesomeEgg/CS386ProjectGroup1.git

1 Introduction

The purpose of this document is to examine the macro structure of the system we are designing as well as the components of the system and how they interact. This document will be heavily visual-based with textual descriptions to support each visual. As a reminder, we are developing a hardware instruction generator for the microfluidic experimentation group at Boston University.

2 System Architecture

In this section, we consider the system-wide architecture. We will first discuss the architecture we have implemented in detail. Visuals will accompany textual descriptions where necessary.

2.1 Architecture Description

We decided that a monolithic architecture would be most applicable and useful for our system. The benefits of the monolithic architecture is cohesion, and ease of communication between components. The drawbacks of the monolithic architecture is coupling, but we have determined that since the system will not be scaled or have new features implemented, the coupling is not negatively impactful. Listed below is the architecture diagram for the system.

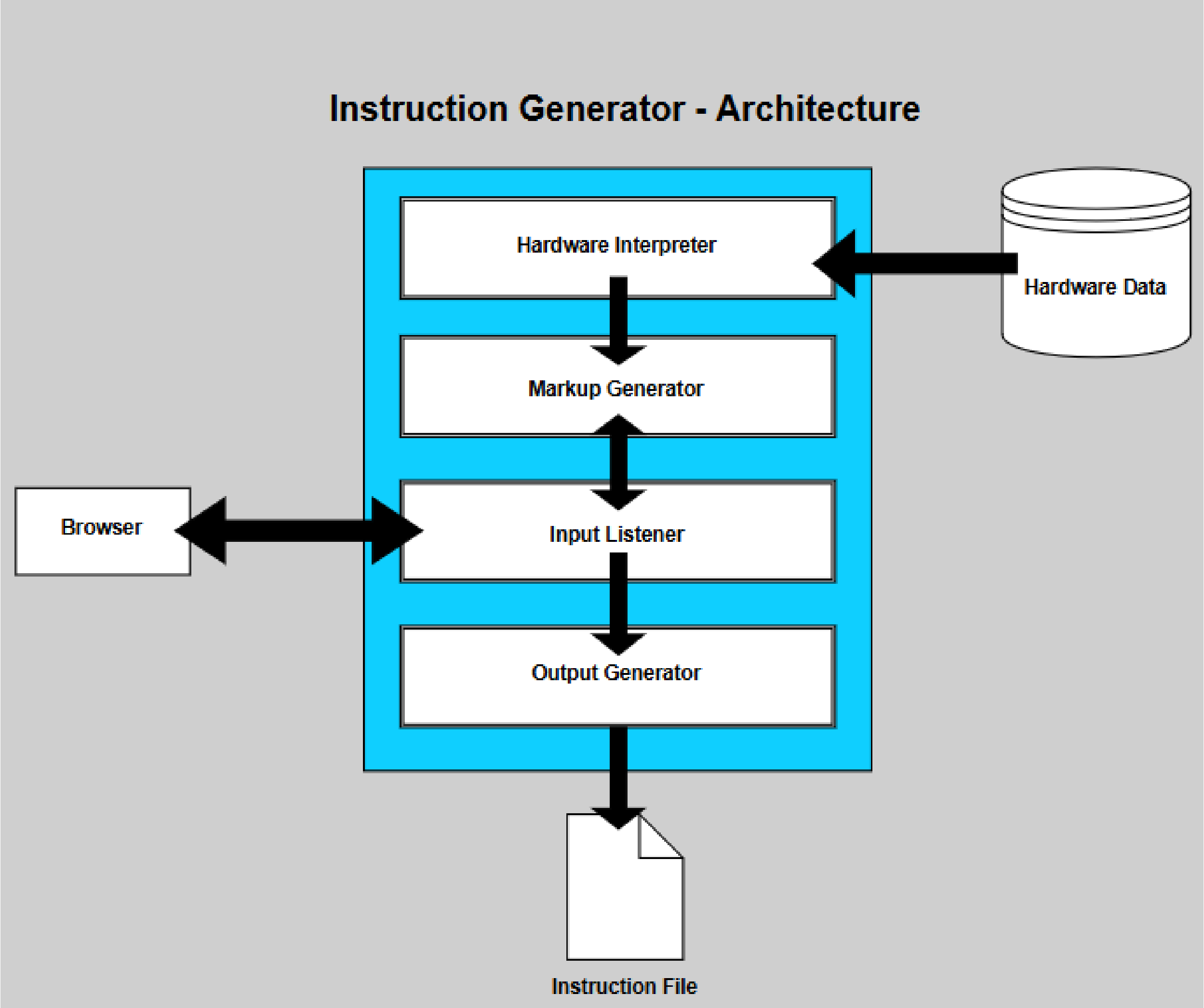

Figure 1:System Architecture

3 System Descriptions

In this section, we consider the interplay of the components that makeup the system architecture. We have included several diagrams that serve to demonstrate functionality, implementation and execution of the system.

3.1 Class Diagram

We have discussed the class diagram for the system in previous documents, but have revised the document to account for variations in features and to better describe pre-existing components. Listed below is the revised class diagram for the system.

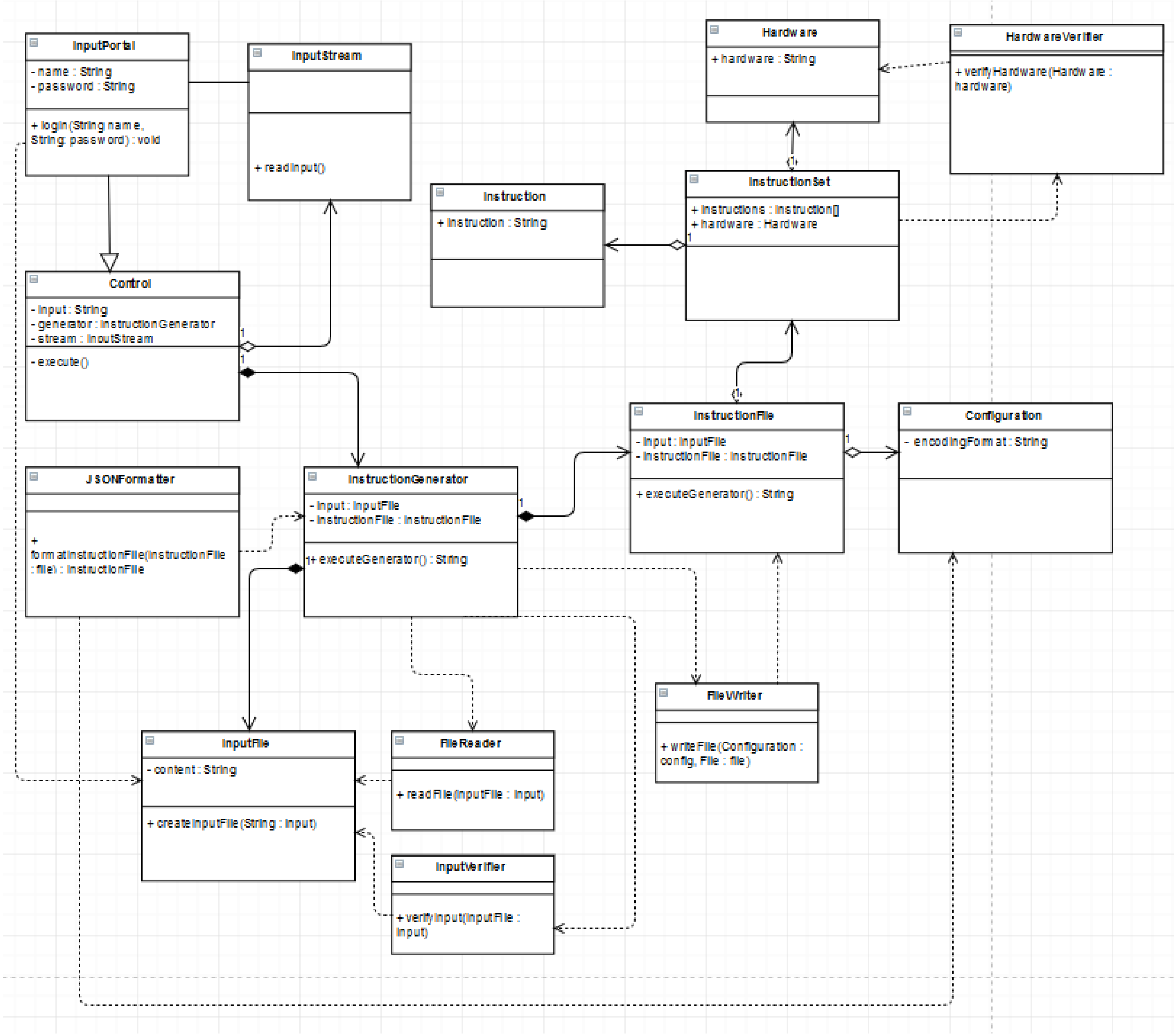
InputPortal
- name : String
- password : String
+ login(String name, String password) : void
InputStream
+ readInput()
Hardware
+ hardware : String
HardwareVerifier
+ verifyHardware(Hardware : hardware)
Instruction
+ instruction : String
InstructionSet
+ instructions : Instruction[]
+ hardware : Hardware
1
Control
- input : String
- generator : InstructionGenerator
- stream : InoutStream
- execute()
1
1
InstructionFile
- input : InputFile
- instructionFile : InstructionFile
+ executeGenerator() : String
1
Configuration
- encodingFormat : String
JSONFormatter
+ formatInstructionFile(InstructionFile : file) : InstructionFile
InstructionGenerator
- input : InputFile
- instructionFile : InstructionFile
1
1+ executeGenerator() : String
FileWriter
+ writeFile(Configuration : config, File : file)
InputFile
- content : String
+ createInputFile(String : input)
FileReader
+ readFile(InputFile : input)
InputVerifier
+ verifyInput(InputFile : input)

Figure 2: Revised Class Diagram

At a glance, it easy to discern that the structure is tightly coupled. This is intended and is a design trade-off for our creation of a monolithic architecture. Tight coupling is often seen as taboo in large systems because of the extra work associated with maintenance and new features, but our project is discretely defined and will not be updated to accommodate new functionality or scalability. That being said, coupling will not be negatively impactful towards our system.

3.2 Sequence Diagram

We will now consider the flow of logic within our system. The logic for our system is relatively simple as the user's need for system is quite explicit. To describe the flow of logic in the system, we have listed the sequence diagram below.

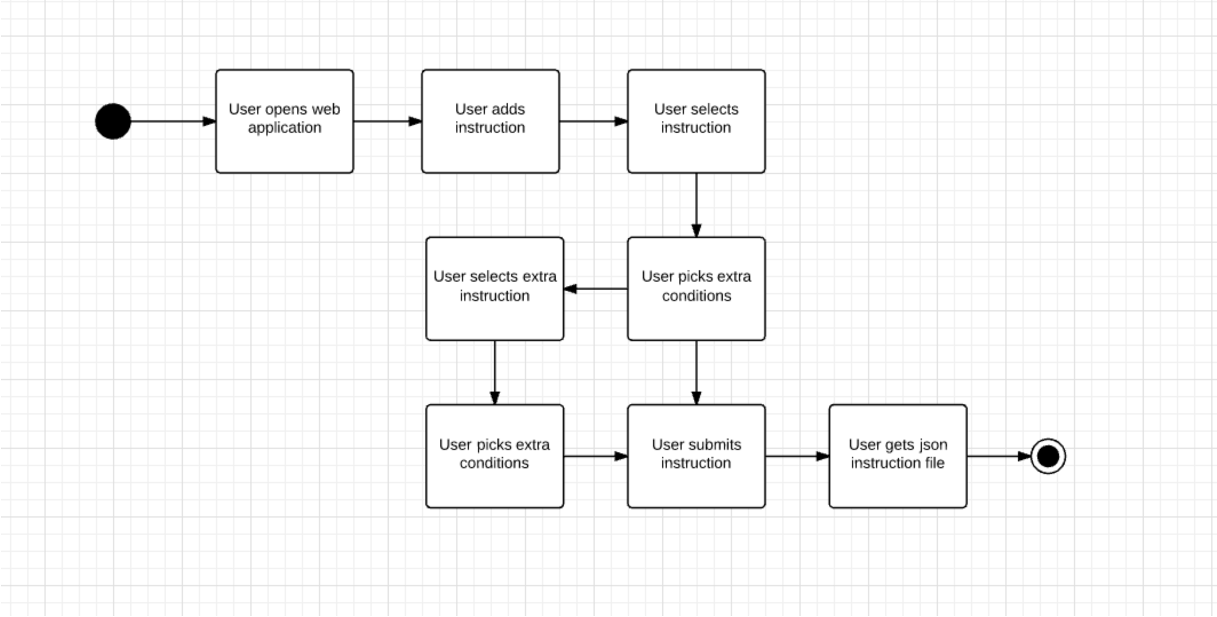
User opens web application
User adds instruction
User selects instruction
User selects extra instruction
User picks extra conditions
User picks extra conditions
User submits instruction
User gets json instruction file

Figure 3: Sequence Diagram

3.3 Design Patterns

Listed below is the design pattern we implemented for the system.

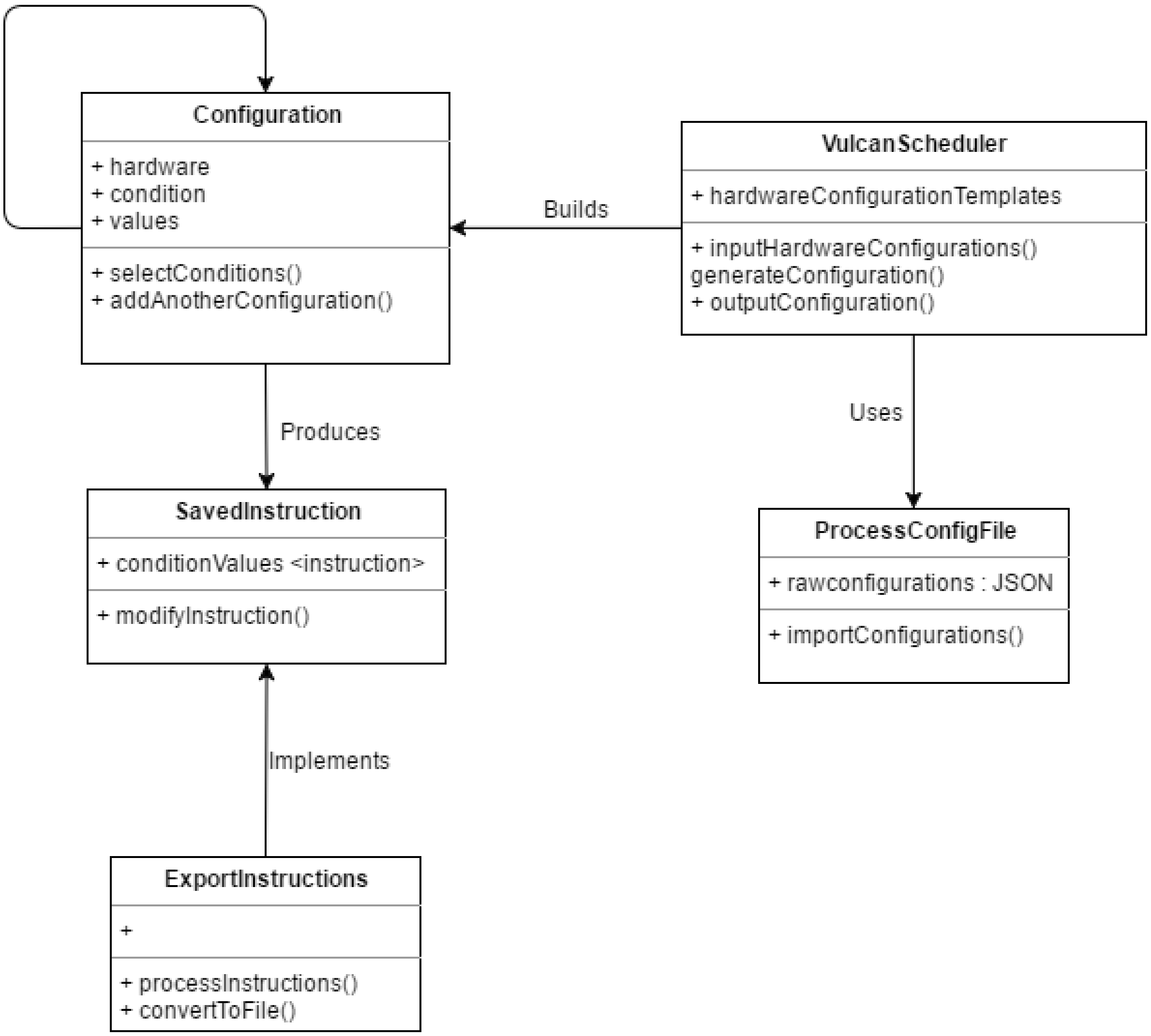

Figure 4: Design Pattern Implementation

4 Conclusion

This concludes the internal design description of the system we are designing. We are satisfied with the current design, as all requirements are satisfied; but, are prepared to make changes as needed.

Group Participation

Listed below is a table containing the group participation weights for each team member.

Team Member	Participation
James Beasley	20%
Charles Beck	20%
Charles Duso	20%
Alexander Grzesiak	20%
Erik Strauss	20%

Table 1: Group Participation Weights